What is claimed is:

1: A method comprising:

processing a group of instructions in topological dependence order;

selecting an instruction, associated with at least one operand, to schedule;

allocating a register to one or more of the operands;

determining if the live range of the operand spans a function call;

if so, attempting to allocate a preserved register;

if not, attempting to allocate a scratch register; and

if the determination is unknown, treating the live range as tentative; and

if the allocation was successful, scheduling the instruction.

2: The method of claim 1, wherein allocating a register to one or more of the operand includes:

determining if the operand's use begins a live range or ends a live range;

if the operand's use begins a live range, attempting to allocate a register;

if the operand's use ends a live range, marking the register associated with the operand as available for reallocation.

3: The method of claim 1, wherein attempting to allocate a preserved register includes:

determining if a preserved register is available for allocation from a list of previously used preserved registers; and

if so, allocating the available preserved register for the live range defined by the operand.

4: The method of claim 3, further including, if a preserved register is not available from the list of previously used preserved registers,

attempting to find a tentative live range that includes a preserved register that is in the list of previously used preserved registers; and

if successful, allocating this preserved register for the live range defined by the operand, and reserving a different preserved register for the tentative live range.

5: The method of claim 4, further including, if the attempt to find a tentative live range that includes a preserved register that is in the list is not successful,

adding a new preserved register to the list of previously used preserved registers; and

allocating this new preserved register to the live range defined by the operand.

6: The method of claim 1, wherein attempting to allocate a scratch register includes:

determining if a scratch register is available for allocation;

if so, allocating a scratch register for the live range defined by the operand;

if not,

attempting to steal a scratch register from a live range marked as tentative;

determining if the attempted theft was successful;

if so, allocating the stolen scratch register to the live range defined by the operand; and

if not, attempting to allocate a preserved register.

7: The method of claim 1, wherein treating the live range as tentative further includes:

if both a scratch register and a preserved register are available for allocation, reserving both the scratch register and the preserved register to the live range;

allowing another instruction to be processed; and

waiting for more information about the live range.

8: The method of claim 7, further including, when more information is available:

if the last use of the live range has been scheduled, allocating the reserved scratch register to the live range defined by the operand, and freeing the reserved preserved register;

if a function call has been scheduled before the last use of the live range or the reserved scratch register has been stolen, allocating the preserved register to the live range defined by the operand and freeing the reserved scratch register.

9: The method of claim 7, wherein reserving a preserved register for the tentative live range includes:

preferring to reserve a currently unused preserved register that is in the list of previously used preserved registers, before reserving a new unused preserved register for the tentative live range.

10: The method of claim 9, wherein treating the live range as tentative further includes:

if a scratch register is not available for allocation, attempting to allocate a preserved register to the live range.

11: The method of claim 1, wherein allocating a register to at least one or more of operands associated with the instruction includes:

if no proper registers are available for allocation, inserting a register spill; and

either re-attempting to allocate a register, or selecting another instruction to schedule.

12: An article comprising:

a machine readable medium having a plurality of machine accessible instructions, wherein when the instructions are executed, the instructions provide for:

processing a group of instructions in topological dependence order;

selecting an instruction, associated with at least one operand, to schedule;

allocating a register to one or more of the operands;

determining if the live range of the operand spans a function call;

if so, attempting to allocate a preserved register;

if not, attempting to allocate a scratch register; and

if the determination is unknown, treating the live range as tentative; and

if the allocation was successful, scheduling the instruction.

13: The article of claim 12, wherein the instructions providing for allocating a register to one or more of the operand includes instructions providing for:

determining if the operand's use begins a live range or ends a live range;

if the operand's use begins a live range, attempting to allocate a register;

if the operand's use ends a live range, marking the register associated with the operand as available for reallocation.

14: The article of claim 12, wherein the instructions providing for attempting to allocate a preserved register includes instructions providing for:

determining if a preserved register is available for allocation from a list of previously used preserved registers; and

if so, allocating the available preserved register for the live range defined by the operand.

15: The article of claim 14, further including instructions providing for, if a preserved register is not available from the list of previously used preserved registers,

attempting to find a tentative live range that includes a preserved register that is in the list of previously used preserved registers; and if successful, allocating this preserved register for the live range defined by the operand, and reserving a different preserved register for the tentative live range.

16: The article of claim 15, further including instructions providing for, if the attempt to find a tentative live range that includes a preserved register that is in the list is not successful,

adding a new preserved register to the list of previously used preserved registers; and

allocating this new preserved register to the live range defined by the operand.

17: The article of claim 12, wherein the instructions providing for attempting to allocate a scratch register includes instructions providing for:

determining if a scratch register is available for allocation;

if so, allocating a scratch register for the live range defined by the operand;

if not,

attempting to steal a scratch register from a live range marked as tentative;

determining if the attempted theft was successful;

if so, allocating the stolen scratch register to the live range defined by the operand; and

if not, attempting to allocate a preserved register.

18: The article of claim 12, wherein the instructions providing for treating the live range as tentative further includes instructions providing for:

if both a scratch register and a preserved register are available for allocation, reserving both the scratch register and the preserved register to the live range;

allowing another instruction to be processed; and

waiting for more information about the live range.

19: The article of claim 18, further including instructions providing for, when more information is available:

if the last use of the live range has been scheduled, allocating the reserved scratch register to the live range defined by the operand, and freeing the reserved preserved register;

if a function call has been scheduled before the last use of the live range or the reserved scratch register has been stolen, allocating the preserved register to the live range defined by the operand and freeing the reserved scratch register.

20: The article of claim 18, wherein the instructions providing for reserving a preserved register for the tentative live range includes instructions providing for:

preferring to reserve a currently unused preserved register that is in the list of previously used preserved registers, before reserving a new unused preserved register for the tentative live range.

21: The article of claim 20, wherein the instructions providing for treating the live range as tentative further includes instructions providing for:

if a scratch register is not available for allocation, attempting to allocate a preserved register to the live range.

22: The article of claim 12, wherein the instructions providing for allocating a register to at least one or more of operands associated with the instruction includes instructions providing for:

if no proper registers are available for allocation, inserting a register spill; and

either re-attempting to allocate a register, or selecting another instruction to schedule.

23: An apparatus comprising:

an instruction scheduler to:

process a group of instructions in topological dependence order, and

select an instruction, associated with at least one operand, to schedule

schedule an instruction, if the register allocation was successful; and

a register allocator to:

allocate register to a live range associated with one or more operand,

determine if the live range of the operand spans a function call,

if so, attempt to allocate a preserved register, if not, attempt to allocate a scratch register, and

if the determination is unknown, treat the live range as tentative.

24: The apparatus of claim 23, wherein the register allocator is capable of:

determining if the operand's use begins a live range or ends a live range;

if the operand's use begins a live range, attempting to allocate a register;

if the operand's use ends a live range, marking the register associated with the operand as available for reallocation.

25: The apparatus of claim 23, wherein the register allocator is capable of, when attempting to allocate a preserved register:

determining if a preserved register is available for allocation from a list of previously used preserved registers; and

if so, allocating the available preserved register for the live range defined by the operand .

26: The apparatus of claim 25, wherein the register allocator is capable of, if a preserved register is not available from the list of previously used preserved registers:

attempting to find a tentative live range that includes a preserved register that is in the list of previously used preserved registers; and

if successful, allocating this preserved register for the live range defined by the operand, and reserving a different preserved register for the tentative live range.

27: The apparatus of claim 26, wherein the register allocator is capable of, if the attempt to find a tentative live range that includes a preserved register that is in the list is not successful:

adding a new preserved register to the list of previously used preserved registers; and

allocating this new preserved register to the live range defined by the operand.

28: The apparatus of claim 23, wherein the register allocator is capable of, when attempting to allocate a scratch register:

determining if a scratch register is available for allocation;

if so, allocating a scratch register for the live range defined by the operand;

if not,

attempting to steal a scratch register from a live range marked as tentative;

determining if the attempted theft was successful;

if so, allocating the stolen scratch register to the live range defined by the operand; and

if not, attempting to allocate a preserved register.

29: The apparatus of claim 23, wherein the register allocator is capable of, when treating a live range as tentative:

if both a scratch register and a preserved register are available for allocation, reserving both the scratch register and the preserved register to the live range;

allowing another instruction to be processed; and

waiting for more information about the live range.

30: The apparatus of claim 29, wherein the register allocator is capable of, when more information is available:

if the last use of the live range has been scheduled, allocating the reserved scratch register to the live range defined by the operand, and freeing the reserved preserved register;

if a function call has been scheduled before the last use of the live range or the reserved scratch register has been stolen, allocating the preserved register to the live range defined by the operand and freeing the reserved scratch register.

31: The apparatus of claim 30, wherein the register allocator is capable of, when reserving a preserved register for the tentative live range preferring to reserve a currently unused preserved register that is in the list of previously used preserved registers, before reserving a new unused preserved register for the tentative live range .

32: The apparatus of claim 31, wherein the register allocator is capable of, when treating a live range as tentative:

if a scratch register is not available for allocation, attempting to allocate a preserved register to the live range.

33. The apparatus of claim 23, the register allocator is capable of:

if no proper registers are available for allocation, inserting a register spill; and

either re-attempting to allocate a register, or selecting another instruction to schedule.

34. The apparatus of claim 23, wherein the register allocator includes:

a scratch register allocator to allocate scratch registers to live ranges that do not span a function call;

a preserved register allocator to allocate preserved registers to live ranges that span a function call; and

a tentative register allocator to allocate either a scratch or a preserved register to live ranges that are not immediately known whether or not they span a function call.

35: A system comprising:

a set of instructions to be compiled;

a set of preserved registers capable of storing values that are to be preserved across function calls;

a set of scratch registers capable to storing values that do not need to be preserved across function calls; and

an integrated compiler having:

an instruction scheduler to:

process a group of instructions in topological dependence order, and

select an instruction, associated with at least one operand, to schedule

schedule an instruction, if the register allocation was successful; and

a register allocator to:

allocate register to a live range associated with one or more operand,

determine if the live range of the operand spans a function call,

if so, attempt to allocate a preserved register,

if not, attempt to allocate a scratch register, and

if the determination is unknown, treat the live range as tentative.

36: The system of claim 35, wherein the register allocator is capable of:

determining if the operand's use begins a live range or ends a live range;

if the operand's use begins a live range, attempting to allocate a register;

if the operand's use ends a live range, marking the register associated with the operand as available for reallocation.

37: The system of claim 35, wherein the register allocator is capable of, when attempting to allocate a preserved register:

determining if a preserved register is available for allocation from a list of previously used preserved registers; and if so, allocating the available preserved register for the live range defined by the operand .

38: The system of claim 37, wherein the register allocator is capable of, if a preserved register is not available from the list of previously used preserved registers:

attempting to find a tentative live range that includes a preserved register that is in the list of previously used preserved registers; and

if successful, allocating this preserved register for the live range defined by the operand, and reserving a different preserved register for the tentative live range.

39: The system of claim 38, wherein the register allocator is capable of, if the attempt to find a tentative live range that includes a preserved register that is in the list is not successful:

adding a new preserved register to the list of previously used preserved registers; and

allocating this new preserved register to the live range defined by the operand.

40: The system of claim 35, wherein the register allocator is capable of, when attempting to allocate a scratch register:

determining if a scratch register is available for allocation;

if so, allocating a scratch register for the live range defined by the operand;

if not,

attempting to steal a scratch register from a live range marked as tentative;

determining if the attempted theft was successful;

if so, allocating the stolen scratch register to the live range defined by the operand; and

if not, attempting to allocate a preserved register.

41: The system of claim 35, wherein the register allocator is capable of, when treating a live range as tentative:

if both a scratch register and a preserved register are available for allocation, reserving both the scratch register and the preserved register to the live range;

allowing another instruction to be processed; and

waiting for more information about the live range.

42: The system of claim 41, wherein the register allocator is capable of, when more information is available:

if the last use of the live range has been scheduled, allocating the reserved scratch register to the live range defined by the operand, and freeing the reserved preserved register;

if a function call has been scheduled before the last use of the live range or the reserved scratch register has been stolen, allocating the preserved register to the live range defined by the operand and freeing the reserved scratch register.

43: The system of claim 42, wherein the register allocator is capable of, when reserving a preserved register for the tentative live range
preferring to reserve a currently unused preserved register that is in the list of previously used preserved registers, before reserving a new unused preserved register for the tentative live range .

44: The system of claim 43, wherein the register allocator is capable of, when treating a live range as tentative:

if a scratch register is not available for allocation, attempting to allocate a preserved register to the live range.

45. The system of claim 35, the register allocator is capable of:

if no proper registers are available for allocation, inserting a register spill; and

either re-attempting to allocate a register, or selecting another instruction to schedule.

46. The system of claim 35, wherein the register allocator includes:

a scratch register allocator to allocate scratch registers to live ranges that do not span a function call;

a preserved register allocator to allocate preserved registers to live ranges that span a function call; and

a tentative register allocator to allocate either a scratch or a preserved register to live ranges that are not immediately known whether or not they span a function call.